Online C/C++ Code Editor – web compiler

A PROJECT REPORT

Submitted by

Shaik Adam Durwaish (21BCS4222)

in Partial Fulfillment of the Requirements for the Degree of

BACHELOR OF ENGINEERING

IN

COMPUTER SCIENCE

Chandigarh University

April - 2025

BONAFIDE CERTIFICATE

Certified that this project report "**Online C/C++ Compiler – Web Compiler**" is the bonafide work of "**Shaik Adam Durwaish**" carried out the project work under my supervision.

SIGNATURE	**SIGNATURE**
Mr. Aman Kaushik	Prof. Ravneet Kaur
HEAD OF THE DEPARTMENT	**SUPERVISOR**
AIT-CSE	AIT-CSE

Submitted for the project viva-voce examination held on 18/05/24.

INTERNAL EXAMINER	**EXTERNAL EXAMINER**

ACKNOWLEDGEMENT

We are very pleased to present this report "Online C/C++ Compiler – Web Compiler". We are deeply indebted to our esteemed guide Prof. Ravneet Kaur of Department of Computer Science for his valuable guidance, excellent guidance and constant encouragement throughout the work.

We thank Dr. Aman Kaushik, HOD for giving us an opportunity to present our project. Many thanks to Prof. Ravneet Kaur for his continued interest in the project and helpful suggestions to resolve the issues encountered during the work. We are also grateful to all the other lecturers and officials of the department who have given their valuable support for their friendly cooperation.

We also thank the librarian for providing useful educational material to CU. Last but not the least, we thank our family members, friends and all our well-wishers for their encouragement and moral support during the project work. We thank God Almighty for his best blessings that he showed me in completing this!

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ABSTRACT

In today's fast-paced digital ecosystem, web-based tools have become increasingly vital for enhancing accessibility and collaboration in programming. Traditional offline compilers require lengthy installations and platform-specific configurations, which can hinder rapid development and learning. This project aims to address those limitations by providing an intuitive and efficient online C/C++ compiler that runs entirely in the browser, making coding possible from virtually any device without the need for a local setup.

The system is built using a modern tech stack comprising **React.js** for the frontend, integrated with **Monaco Editor** for code editing and **Xterm.js** for terminal output. The backend is handled by **Node.js** and **Express**, where user-submitted C/C++ code is compiled and executed in real time through system-level compilers like `g++`. The platform supports runtime input, which enhances interactivity and allows users to test logic-based and input-dependent programs effectively. To ensure user safety and system integrity, the execution is sandboxed, preventing any malicious access or unintended system calls.

This project demonstrates how web technologies can be combined to deliver a functional development environment for compiled languages like C and C++. It not only serves as a useful tool for students and developers but also showcases the potential of hybrid cloud-based IDEs. With its responsive design and real-time capabilities, the compiler is suitable for educational platforms, coding practice, and technical interviews, especially in remote and cross-platform scenarios.

Keywords:

C/C++ Compiler, Online IDE, Web-based Code Editor, Node.js, React.js, Monaco Editor, Xterm.js, Real-time Execution, Code Runner, Sandbox Environment

ABBREVIATIONS

MERN - MongoDB, Express.js, React.js, Node.js

API - Application Programming Interface

UI - User Interface

UX - User Experience

CRUD - Create, Read, Update, Delete

JSON - JavaScript Object Notation

DBMS - Database Management System

REST - Representational State Transfer

URL - Uniform Resource Locator

HTML - Hypertext Markup Language

CSS - Cascading Style Sheets

SQL - Structured Query Language

CDN - Content Delivery Network

HTTPS - Hypertext Transfer Protocol Secure

MVC - Model-View-Controller

SSL - Secure Sockets Layer

JWT - JSON Web Token

CHAPTER 1:

INTRODUCTION

In the evolving world of software development and programming education, accessibility and ease of use are key to accelerating learning and productivity. Traditionally, C and C++ programs are written and executed using offline tools that require proper setup of compilers and environments, which can be a barrier—especially for beginners and those working on shared or limited-resource systems. Recognizing this challenge, there is a growing demand for browser-based development platforms that eliminate these complexities and offer instant code execution capabilities directly through the web.

Recent advancements in web technologies and cloud computing have made it possible to build real-time code execution environments that function seamlessly in the browser. These platforms promote a more inclusive and flexible approach to coding by allowing users to write, compile, and execute code from any device, regardless of its specifications or installed software. While online IDEs exist for scripting languages like Python and JavaScript, providing a responsive and secure environment for compiled languages like C and C++ presents unique challenges in terms of compilation, runtime input handling, and sandboxing for secure execution.

To address these technical gaps, this report introduces the development of a web-based **Online C/C++ Compiler** built using the **MERN stack (MongoDB, Express.js, React.js, Node.js)** along with **Monaco Editor** and **Xterm.js** for a seamless coding experience. The platform enables users to write C/C++ code, provide runtime input, and view real-time terminal output—all within the browser. Code is sent to the server,

compiled using system-level compilers (`gcc/g++`), and the results are displayed back to the user with minimal delay. This project outlines the overall system architecture, security considerations, and technical implementation while highlighting the project's potential to serve as an educational tool, a practice environment for programmers, and a foundation for future development of full-scale online IDEs for compiled languages.

By combining real-time capabilities with a user-friendly interface, the Online C/C++ Compiler bridges the gap between traditional development environments and the need for accessible, modern coding solutions. Its design emphasizes simplicity, speed, and reliability, making it ideal for both learning and practical use. As coding becomes an essential skill across disciplines, tools like this play a crucial role in democratizing programming access, reducing entry barriers, and enabling a smoother transition for beginners into the world of system-level languages like C and C++.

1.1 PROBLEM STATEMENT

1.1.1 Problem Definition

Maintaining a balanced micronutrient intake is crucial for overall health, yet many people face challenges in monitoring and achieving optimal levels of essential vitamins and minerals. Micronutrients like vitamin D, iron, magnesium, and zinc play vital roles in immune function, energy metabolism, and disease prevention, but inadequate or excessive intake can lead to health issues. Existing diet-tracking tools often lack comprehensive features for tracking these nutrients, focusing primarily on macronutrients (carbohydrates, proteins, fats) instead. This lack of detail in micronutrient tracking prevents users from effectively managing their intake and understanding nutrient gaps in their diets, leaving a significant gap in current nutrition management solutions.

Moreover, the popularity of high-dose vitamin supplements has led to increased consumption without adequate monitoring or understanding of potential risks. Studies, especially those focused on HIV patients undergoing highly active antiretroviral therapy (HAART), have shown that high-dose vitamin supplementation does not yield improvements in health outcomes compared to standard doses and may, in fact, pose risks by increasing liver enzyme levels and other markers of toxicity. This highlights a broader issue where synthetic supplements are often used as a quick remedy for nutrient deficiencies, potentially neglecting the benefits of natural nutrient sources and balanced intake.

Addressing these gaps requires a tool that not only provides detailed micronutrient tracking but also educates users on sourcing nutrients from natural, whole foods and encourages balanced intake rather than reliance on supplements.

1.1.2 Problem Overview

Learning and working with C and C++ requires a proper development environment that includes installing compilers, setting up paths, and configuring editors, which can be a significant hurdle for beginners, students, and even professionals working across different systems. Many users struggle with environment setup issues, system incompatibilities, or lack access to required administrative rights on shared machines, ultimately delaying their learning or workflow. While desktop IDEs offer robust features, they lack portability and demand considerable system resources, making them unsuitable for quick coding sessions, low-end devices, or educational setups like computer labs with restricted access.

Furthermore, traditional compilation and execution methods provide limited flexibility in remote learning environments and are not well-suited for online assessments or collaborative development. Although some online compilers exist, many suffer from limitations such as lack of runtime input support, slow execution, or outdated user interfaces. These limitations reduce the effectiveness of such tools for solving algorithmic problems, testing input-output-based logic, or practicing competitive programming, especially for compiled languages where runtime behavior is critical.

To address these challenges, there is a need for a lightweight, responsive, and feature-rich online compiler that allows users to write, compile, and execute C/C++ code directly in the browser with support for runtime inputs, terminal-like output, and secure, real-time execution. Such a tool would remove entry barriers, provide an accessible environment for practice and learning, and serve as a powerful resource for both individuals and educational institutions aiming to teach or assess C/C++ programming without infrastructure constraints.

In addition to accessibility challenges, another major issue lies in the inability of many online platforms to offer a seamless, real-time coding experience. Most existing compilers lack modern development features such as syntax highlighting, input-based execution, and interactive output visualization, which are essential for an effective coding environment. Beginners often find these tools confusing or underwhelming, leading to frustration and disengagement. Moreover, platforms that do offer advanced features are often behind paywalls, have usage limitations, or are designed for multiple languages, which dilutes the focus from the specific needs of C and C++ programmers. This lack of a dedicated, free, and interactive online environment specifically tailored for C/C++ limits the learning curve and experimentation for users who wish to dive deep into system-level programming.

Security is another critical concern when executing user-submitted code on a remote server. Running C/C++ code poses higher risks compared to interpreted languages due to the potential for low-level operations and memory access. Without proper sandboxing or isolation mechanisms, such platforms are vulnerable to exploitation, misuse, or system crashes. Many existing platforms do not provide clear information on how user code is executed or fail to restrict potentially harmful code behavior, making them unreliable or unsafe for open usage. A secure, sandboxed execution model is vital to protect both the server infrastructure and the integrity of the platform, especially when targeting a broad audience that includes students, educators, and freelance developers.

1.2 SPECIFICATIONS

1.2.1 Hardware Specifications

To ensure the smooth development, testing, and deployment of the Online Code Editor application, the following hardware components are used by us:

1. Development Machine:
    - Processor: Intel i5 or higher / AMD Ryzen 5 or higher
    - RAM: 8 GB minimum (16 GB recommended for handling multiple applications simultaneously)
    - Storage: SSD with at least 256 GB for faster performance
    - Graphics: Integrated graphics are sufficient for basic web development; however, a dedicated GPU may be beneficial for handling data visualization tools.
    - Network: Reliable internet connection for API integrations, data retrieval, and deployment

2. Server Specifications:
    - Processor: Intel Xeon or equivalent for handling multiple requests and ensuring faster response times
    - RAM: 16 GB or more to handle concurrent user activity efficiently
    - Storage: 500 GB SSD for database storage and scalability
    - Network: High-speed network with redundancy to ensure minimal downtime for users

1.2.2 Software Specifications

The software stack for this project is composed of both backend and frontend components, alongside additional tools for version control, testing, and deployment.

1. Operating System:
    - Development Environment: Windows 11
    - Server Environment: Linux (Ubuntu Server 20.04 or higher recommended for stability and security)

2. Development Tools:
    - Code Editor: Visual Studio Code (VS Code) for efficient code writing and debugging, with extensions for Node.js and React.js
    - Version Control: Git for tracking code changes, with GitHub as the repository host
    - Testing Tools: Postman for API testing; Jest and Mocha for unit and integration tests

3. Frontend:
    - React.js: JavaScript library for building interactive UI components and managing state
    - HTML5 and CSS3: For basic structure and styling of the application interface
    - Bootstrap or Material-UI: For responsive, user-friendly component design

4. Backend:
    - Node.js: JavaScript runtime environment for building scalable server-

side applications

- Express.js: Web framework for managing server operations and API endpoints

5. Database:
    - MongoDB: NoSQL database for efficient data storage and quick retrieval of user data.
    - Mongoose: Object Data Modeling (ODM) library for MongoDB to define schemas and interact with MongoDB seamlessly

6. External APIs:
    - Nutrition APIs: To retrieve accurate, real-time syncing information for various libraries
    - Authentication APIs: For secure user authentication (e.g., OAuth for social logins)

7. Deployment:
    - Hosting: Cloud services like Heroku, AWS, or DigitalOcean for reliable, scalable deployment of the application
    - Continuous Integration/Continuous Deployment (CI/CD): GitHub Actions for automated testing and deployment

These specifications provide a robust foundation for the online Code Editor application, supporting efficient development, user-friendly interface, and secure, scalable deployment.

CHAPTER 2:

LITERATURE SURVEY

With the rise of digital learning and the increasing popularity of online development environments, the demand for web-based compilers has surged, especially among students, educators, and developers who seek accessible and platform-independent coding tools. Among programming languages, C and C++ hold a critical place due to their close-to-hardware nature and use in system-level programming, operating systems, and performance-intensive applications. Traditional compiler tools require local setup, system configuration, and complex installations, which can be a barrier for beginners and non-technical users. As a result, web-based compilers have emerged as an efficient solution for executing code without the hassle of installation, offering instant feedback and real-time results directly through the browser.

For example, tools like **Replit**, **OnlineGDB**, and **JDoodle** have set the precedent for online compiler platforms. These tools offer users the ability to write and execute C/C++ code from any device with internet access. Replit, for instance, not only supports collaborative coding but also enables code sharing, which enhances peer-to-peer learning. However, most of these tools are general-purpose and support a wide range of languages, often compromising on features that would benefit a focused C/C++ learning experience. Moreover, limitations such as lack of input/output flexibility, session timeouts, or restrictions in file access highlight the need for more refined, dedicated tools that address these shortcomings. These observations reflect a growing requirement for specialized web compilers that can simulate a more authentic C/C++ development environment online.

A comparative study by **Akhtar and Shahid (2021)** evaluated the usability and functionality of various online compilers in educational contexts. The research showed that while these platforms significantly reduce onboarding friction for learners, many lack robust error-handling, code suggestions, and support for user input during runtime. These limitations reduce their effectiveness for teaching algorithmic problem-solving and debugging skills. Furthermore, most platforms do not include sandboxed execution or secure runtime environments, which poses security challenges when executing potentially unsafe code. As such, the paper recommends creating more secure and interactive platforms with input handling, syntax-aware highlighting, and efficient error reporting to enhance the learner experience and safety in online C/C++ execution environments.

In another project, a lightweight online C compiler was implemented specifically for use in educational settings, targeting students preparing for coding competitions and technical interviews. Built using Node.js and Docker containers, the platform allowed multiple concurrent users to compile and execute C code with real-time results and persistent user sessions. The project showed promising results in reducing setup time and improving user engagement in coding practice sessions. However, the report emphasized the importance of UI/UX design in compiler usability, noting that responsive interfaces and real-time feedback mechanisms played a crucial role in keeping students engaged. It also pointed out the need for intuitive output formatting and better integration of input prompts, which are often absent or underdeveloped in basic online compilers.

Finally, in a study exploring the integration of compilers into learning management systems (LMS), researchers introduced a plugin-based online compiler

to support live code execution within e-learning platforms such as Moodle and Canvas. This integration allowed educators to assign coding tasks and receive compiled outputs directly through the LMS. The experiment increased student interactivity and streamlined assessment workflows, but it also revealed issues with latency, limited scalability, and code execution isolation. Based on the findings, the authors concluded that while online compilers provide invaluable utility in academic settings, their effectiveness depends heavily on backend architecture, server optimization, and proper sandboxing to prevent malicious code execution.